# Tidying Up SSLUtils 🧹 (#5040)

ATS Spring 2019 Summit

Masaori Koshiba (masaori@apache.org)

## Original Motivation

Extend loading ssl_multicert.config for QUIC. (#5037)

# “Utils” could be anything

- SSLStats
- SSLDiags
- Wrappers of SSL library APIs
- Loading ssl_multicert.config
  - building list of SSL_CTX
  - handling knobs of ssl_multicert.config
- Initializing SSL_CTX
  ( exposed by TS API )
- TLS Extensions Support
  - TLS Session Ticket/Cache
- Initializing SSL libraries
- Multiple OpenSSL Version support
- BoringSSL / LibreSSL support

…etc

# “Utils” could be anything

- ~~SSLStats~~
- ~~SSLDiags~~
- Wrappers of SSL library APIs
- Loading ssl_multicert.config
  - building list of SSL_CTX
  - handling knobs of ssl_multicert.config

→ SSLMulticertConfigLoader (#5032)

- Initializing SSL_CTX
  ( exposed by TS API )
- TLS Extensions Support
  - ~~TLS Session Ticket/Cache~~
- Initializing SSL libraries
- Multiple OpenSSL Version support
- BoringSSL / LibreSSL support

…etc

# Minimum OpenSSL Version

- ATS 8.x requires OpenSSL 0.9.4+
  - OpenSSL 0.9.4 ( August 9, 1999 )
  - Support until Sep. 2020 (EOL of ATS 8.x)
- Discussed on ML (dev@trafficserver.a.o)
  - ATS 9.0.0 requires OpenSSL 1.0.2+
  - #5074 ( +64 –265 lines )
  - Drop CentOS 6 and Ubuntu 14.04 support

# Common Mistake #1

**build/crypto.m4**

```
AC_DEFUN([TS_CHECK_CRYPTO_OCSP], [
  ...
  AC_SUBST(use_tls_ocsp)
])
```

**include/tscore/ink_config.h.in**

```
#define TS_USE_TLS_OCSP @use_tls_ocsp@
```

**include/tscore/ink_config.h**

```
#define TS_USE_TLS_OCSP 0
```

**iocore/net/SSLUtils.cc**

```
#ifdef TS_USE_TLS_OCSP
...
#else
...
#endif /* TS_USE_TLS_OCSP */
```

# Common Mistake #1

build/crypto.m4

```
AC_DEFUN([TS_CHECK_CRYPTO_OCSP], [
  ...
  AC_SUBST(use_tls_ocsp)
])
```

include/tscore/ink_config.h.in

```
#define TS_USE_TLS_OCSP @use_tls_ocsp@
```

include/tscore/ink_config.h

```
#define TS_USE_TLS_OCSP 0
```

iocore/net/SSLUtils.cc

```
#ifdef TS_USE_TLS_OCSP
...
#else
...
#endif /* TS_USE_TLS_OCSP */
```

```
#if TS_USE_TLS_OCSP
```

## Common Mistake #2

`AC_CHECK_HEADERS(header-file)` autoconf macro defines `HAVE_header-file`.

**iocore/net/SSLUtils.cc**

```
#if HAVE_OPENSSL_HMAC_H
#include <openssl/hmac.h>
#endif
```

# Common Mistake #2

`AC_CHECK_HEADERS(header-file)` autoconf macro defines `HAVE_header-file`.

**iocore/net/SSLUtils.cc**

```
#if HAVE_OPENSSL_HMAC_H
#include <openssl/hmac.h>
#endif
```

`AC_CHECK_HEADERS(openssl/hmac.h)` is not in configure.ac nor build/crypto.m4

## Common Mistake #3

Call `SSL_CTX_set_*` functions twice (#5038)

→ Rule: Call functions in in `init_server_ssl_ctx()`

**SSLInitServerContext()**

```
if (SSLConfigParams::ssl_ocsp_enabled) {
  Debug("ssl", "SSL OCSP Stapling is enabled");
  SSL_CTX_set_tlsext_status_cb(ctx, ssl_callback_ocsp_stapling);
```

**ssl_store_ssl_context()**

```
if (SSLConfigParams::ssl_ocsp_enabled) {
  Debug("ssl", "SSL OCSP Stapling is enabled");
  SSL_CTX_set_tlsext_status_cb(ctx, ssl_callback_ocsp_stapling);
```

# Future Plan

- More cleanups
    - Replace the Tokenizer with a stringview/textview based parser.
    - Spin out TLS Session Ticket/Cache
    - Slice SSLMultiCertConfigLoader::init_server_ssl_ctx()
      - e.g. session cahce/tickets, alpn, sni …
- YAML Support

# Open Issues

- Drop NPN support on 9.0.0?
  - OpenSSL 1.0.2 has ALPN
- I_ or P_ prefix for header files are still valid?
- ssl namespace?
  - SSL has many callback functions, some of them are declared in global namespace with *ssl*_ prefix.
  - *ssl* namespace is defined in P_SSLUtils.h
- static function v.s. unnamed namespace

## Thanks

- ⛱️ SSLUtils Cleanups #5040
  https://github.com/apache/trafficserver/issues/5040

# Functions for SSL

```
P_OCSPStapling.h:
void ssl_stapling_ex_init();
bool ssl_stapling_init_cert(SSL_CTX *ctx, X509 *cert, const char *certname);
int ssl_callback_ocsp_stapling(SSL *);

P_SSLCertLookup.h:
ssl_ticket_key_block *ticket_block_alloc(unsigned count);
ssl_ticket_key_block *ticket_block_create(char *ticket_key_data, int ticket_key_len);
ssl_ticket_key_block *ssl_create_ticket_keyblock(const char *ticket_key_path);

P_SSLUtils.h:
ssl_error_t SSLWriteBuffer(SSL *ssl, const void *buf, int64_t nbytes, int64_t &nwritten);
ssl_error_t SSLReadBuffer(SSL *ssl, void *buf, int64_t nbytes, int64_t &nread);
ssl_error_t SSLAccept(SSL *ssl);
ssl_error_t SSLConnect(SSL *ssl);

ProxyProtocol.h:
extern bool ssl_has_proxy_v1(NetVConnection *, char *, int64_t *);

SSLDynlock.h:
extern struct CRYPTO_dynlock_value *ssl_dyn_create_callback(const char *file, int line);
extern void ssl_dyn_lock_callback(int mode, struct CRYPTO_dynlock_value *value, const char *file, int line);
extern void ssl_dyn_destroy_callback(struct CRYPTO_dynlock_value *value, const char *file, int line);

SSLSessionTicket.h:
void ssl_session_ticket_free(void *, void *, CRYPTO_EX_DATA *, int, long, void *);
int ssl_callback_session_ticket(SSL *, unsigned char *, unsigned char *, EVP_CIPHER_CTX *, HMAC_CTX *, int);
```